Our Docket No.: 042390.P6602

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

Yount, C.	Examiner: Kendall, C.
Application No.: 09/475,526	**Art Group: 2122**
Filed: December 30, 1999	
For: A Method and Apparatus for Generation of Validation Tests	

Commissioner of Patents
P.O. Box 1450
Alexandria, VA 22313-1450

DECLARATION UNDER 37 CFR 1.131 IN SUPPORT OF PRIOR INVENTION

Sir :

I, Charles R Yount declare:

1. I am an inventor of the claims of the above-captioned patent application ("the Application") and the inventor of the subject matter described therein.

2. At least prior to June 3, 1997, the filing date of U.S. Patent No. 6,347,388 cited in an Office Action mailed September 10, 2003, the invention claimed in the Application had been conceived and reduced to practice in the United States.

3. Attached Exhibit A, dated April, 1997, is an invention disclosure form describing the design of the Method and Apparatus for Generation of Validation Tests.

4. Exibit A discloses using search techniques which rely on interactive improvement, and use functional coverage as a feedback mechanism to guide these techneques. The interative search techniques navigate through a space of potential solutions by evaluating a subset of those solutions, and then using those evaluations to choose new solutions until an optimal solution is found. The framework of this method includes a test generator (TG), a test analyzer (TA) with coverage monitors, and a feedback engine (FE). See Exibit A at page 1, paragraphs 4-7. This basic flow is given in the following psuedo-code:

```
        Generate or otherwise acquire initial test(s).
        Evaluate these initial test(s) based on validation coverage.
        While (coverage goal is not reached) {
                Use iterative search algorithm in FE to determine next point
    solution space.
                Use TG to generate a new test representing this point.
                Use TA to analyze test for validation coverage.
        }
        (See Exibit A at page 2)
```

5. Therefore, Exibit A establishes that the subject matter claimed in the Application had been reduced to practice in the United States prior to June 3, 1997.

I further declare that all statements made herein of our own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application of any patent issuing thereon.

Dated: Aug. 20, 2004

Charles R. Yount

Method for the Automatic Generation of High-Coverage Validation Test Suites

Charles Yount & Melvyn Goveas

Over the past few years, functional validation has become one of the most critical activities associated with Intel's (and our competitors') success. As our design complexities have increased consistently over time, along with our design sizes, it has become very clear that our older, primarily manual validation strategies have run out of steam.

While automatic test generation has been in use for a few years, this has mostly been limited to random test generation. Only lately have we started to focus a considerable amount of effort on more directed methods. One such method, referred to as "specification-based validation," generates exhaustive sets of tests based on an input specification of the design to be tested. In some sense, it is the opposite of random test-generation. However, the completeness of the tests generated by tools employing this strategy is directly limited by the completeness of the input specification, which can take a considerable amount of manual work to generate; while this form has found good use for architectural-level validation (where the input specification is more-or-less stable over many generations), it becomes quite difficult to employ at lower levels (microarchitectural validation, unit validation), where the majority of the bugs are known to be found. This is because the specification at this level is even more detailed, and can change significantly from one design proliferation to the next.

We have come up with a unique solution that addresses this situation. By employing iterative search techniques to guide an automatic test generator, we believe we can significantly reduce the need for detailed specification input at the lower levels of the design. By instrumenting the design with coverage monitors, the tool is able to "learn its way around the design," so to speak, and instruct the test generator to generate tests that will cover logic that has not yet been tested.

We would like to patent our method of generating high-coverage validation test suites that has the following unique characteristics:

1. The use of search techniques which rely on iterative improvement.
2. The use of functional coverage as a feedback mechanism to guide these techniques.

These two major concepts are discussed in more detail below.

Search techniques: The basic idea behind iterative search techniques is to navigate through a space of potential solutions by evaluating a subset of those solutions and using those evaluations to choose new solutions until an optimal solution is found. There is a large number of these algorithms in use in industry and academia. Examples include genetic algorithms, genetic programming, simulated annealing, neural-net training, and standard hill-climbing approaches. Since some of these are only slight variations of others, it is important not to limit this invention to one particular technique.

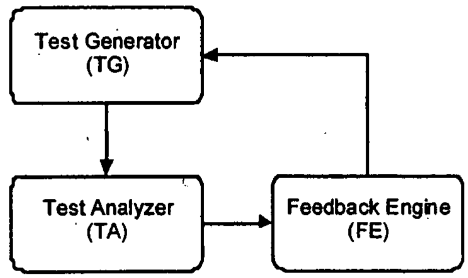
Coverage feedback: The second major characteristic of our invention is the use of coverage as the evaluation criteria to guide the iterative improvement. Otherwise, it is difficult to measure objectively how good a partiular functional test is. Again, there are many kinds of coverage measurements, this invention only requires that the coverage be measureable.

The concept we would like to patent is embodied in a framework that includes a test generator (TG), a test analyzer (TA) with coverage monitors and a feedback engine (FE). This basic flow is given in the following pseudo-code:

EXHIBIT A

erate or otherwise acquire initial test(s).
aluate these initial test(s) based on validation coverage.
While (coverage goal is not reached) {
    Use iterative search algorithm in FE to determine next point in solution space.
    Use TG to generate a new test representing this point.
    Use TA to analyze test for validation coverage.
}

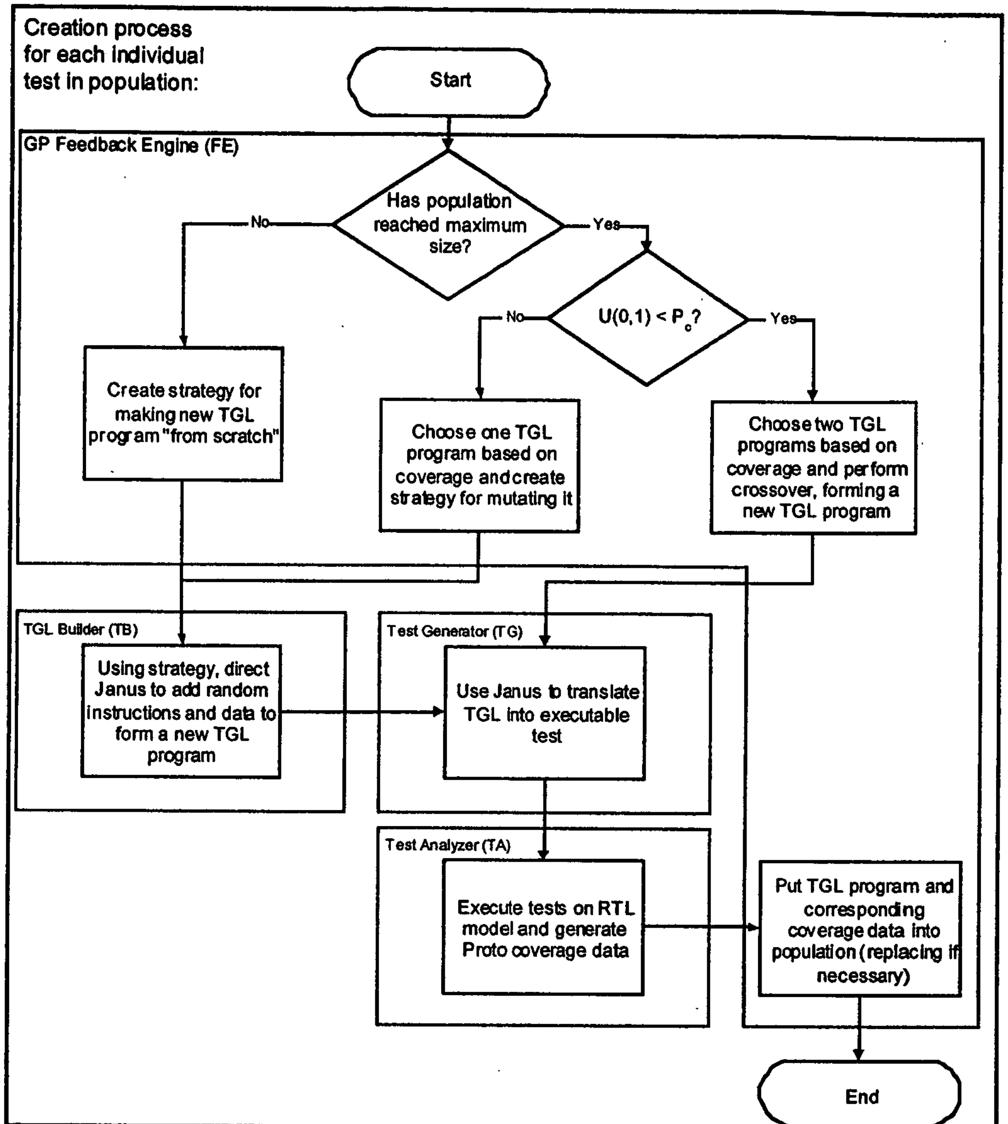
mplified data-flow diagram is shown below.



us different types of test generators might be employed, ranging from random test generators pletely directed test generators (including those employed for specification-based validation ies). Various forms of coverage monitors may be used, ranging from manually-generated event monitors (including silicon-based monitors) to automatically-generated generic monitors to code coverage monitors. Finally, the test-execution medium can vary from based simulation to hardware-accelerated simulation to actual silicon.

ation Technology) are currently executing a program called AutoCov, which defines the described above and four different implementations of this framework (generated by different framework components), targeted at different validation areas, e.g., validation, pre-silicon micro-architectural validation, and pre-silicon unit-level e have completed and demonstrated a proof-of-concept for the first of these ons. Katmai, Merced and Willamette are planning to use these tools.

As a concrete example, one of these implementations is targeted for pre-silicon micro-architectural validation. For this tool, the feedback engine is based on genetic programming, a sub-field of genetic algorithms; the search is thus based on modeling the principles of natural selection. Points in the solution space are represented by a special-purpose Test Generation Language (TGL). Tests are created, evolved or weeded out over time based on coverage; the result is an automatically-generated high-coverage test suite. The feedback engine guides a random test generator, Janus, and the tests will be executed on a software simulator. Coverage is measured using RTL event coverage monitors. Shown is a flowchart of the prototype system; this process is executed in parallel for each test on distributed workstations until the coverage goal has been met. The other three implementations change some or all of these components to perform other types of validation.



We believe that this is the first use of this combination of technologies in the world, and it will be very strategic for Intel to patent this innovation. There certainly have been other uses of iterative search algorithms, and some have been patented, but we know of no other application for generating functional tests. Also, we know of no other use of functional coverage as a mechanism to guide such algorithms.

This technology would be useful to any other company performing functional validation on digital systems, including, but not limited to microprocessors. It could also be used used to generate functional test suites for software systems. Since the method used to generate tests is not detectable in the final product, infringement could not be detected in any way by examining our compeditors' products. Infringement could only be detected with knowledge of a company's test-generation methodology.

Witness:

________________________________ Date: ____________________